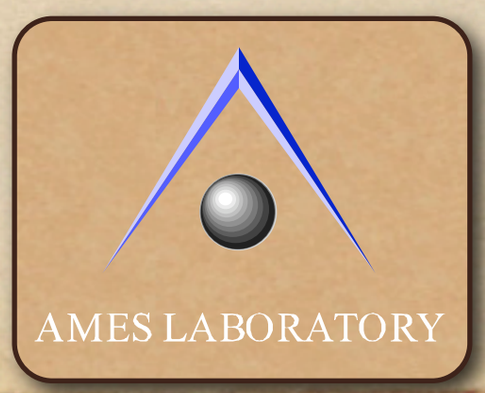
Trading Memory for Disk Using Parallel Access to Fast InfiniBand Disk Arrays for Large Computational Chemistry Applications

Kyle Schochenmaier, Troy Benjegerdes and Brett M. Bode
Scalable Computing Laboratory
Ames Laboratory, U.S. DOE
Iowa State University



Outline

- Problem Statement
- Hardware Configuration
- Software Configuration
- Software Tools
- Results
- Conclusions

Problem Statement

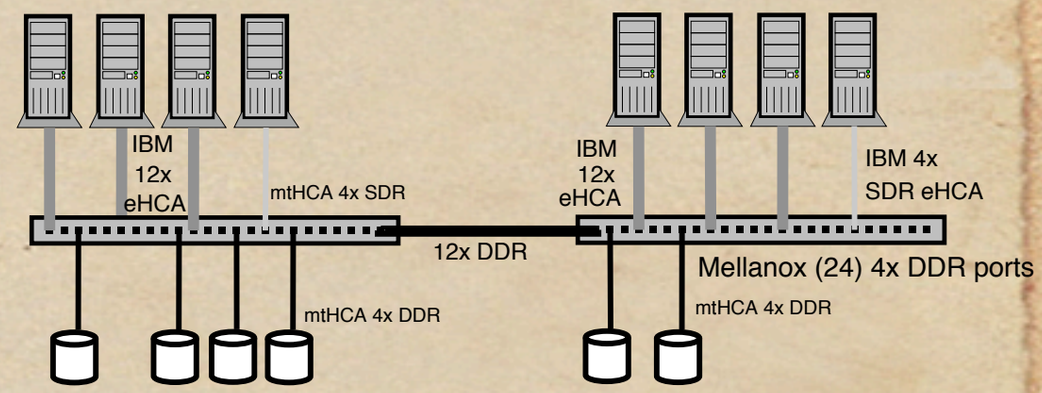
- Our primary application, the GAMESS quantum chemistry app. has many code paths that are quite I/O bound writing and especially reading large temporary storage.
- Most HPC systems are moving towards providing minimal locally attached secondary storage with a corresponding meager I/O bandwidth.
  - This is particularly troubling as the number of CPUs per node is increased.

Proposed Solution

- Network interconnects have reached the point where they can potentially deliver access to secondary storage faster than locally attached storage subsystems.
- This also requires scalable client/server software capable of delivering very high bandwidth to a single node while simultaneously scaling to large numbers of clients.
  - We have chosen to use PVFS2 on Linux clients and servers interconnected by InfiniBand.

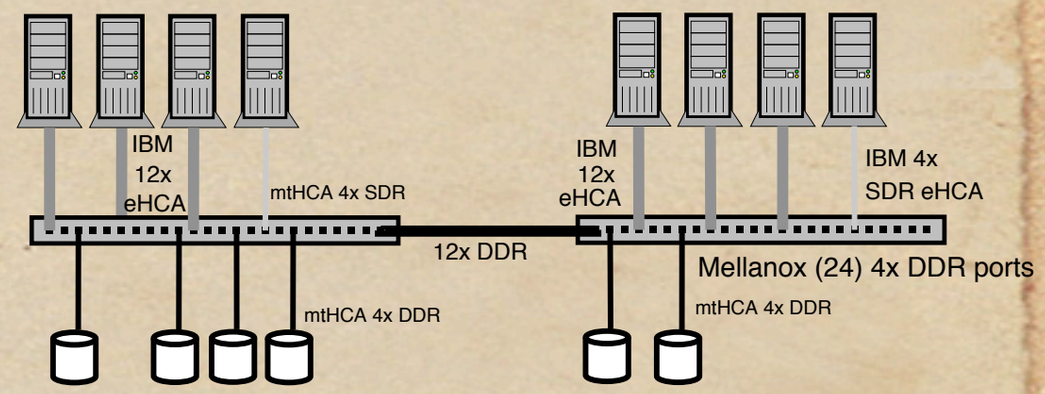
Hardware Configuration

- Six storage servers
  - dual AMD Opteron processors
  - 4 GB RAM
  - 2 Areca PCI-X SATA RAID controllers
  - 16 250 GB Seagate SATA HDs
  - Mellanox 4X DDR PCI-Express InfiniBand adapter (16 Gbps)



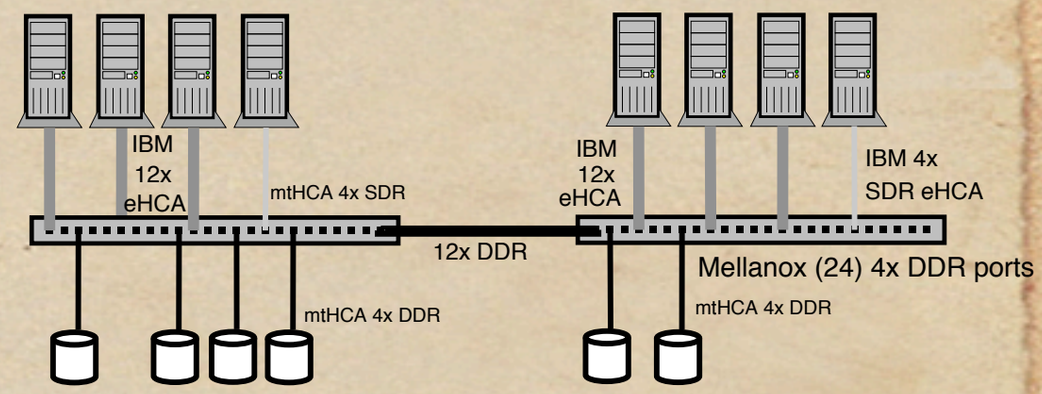
Hardware Configuration

- Eight compute clients
  - quad Power5 processors
  - 8 or 16 GB RAM
  - IBM 12X GX processor bus attached (eHCA) InfiniBand adapter (24 Gbps)
- Interconnect
  - 2 Mellanox 24 port (4X SDR/ DDR) switches
  - Connected together with a 12X DDR link (48 Gbps max data payload)



Software Configuration

- AMD64 version of Debian Linux on storage servers
- PPC64 version of Debian Linux on IBM power5 clients
- PVFS2 running on OpenIB verbs natively. Version 1.5.1 ++ (from latest development tree)



NetPIPE

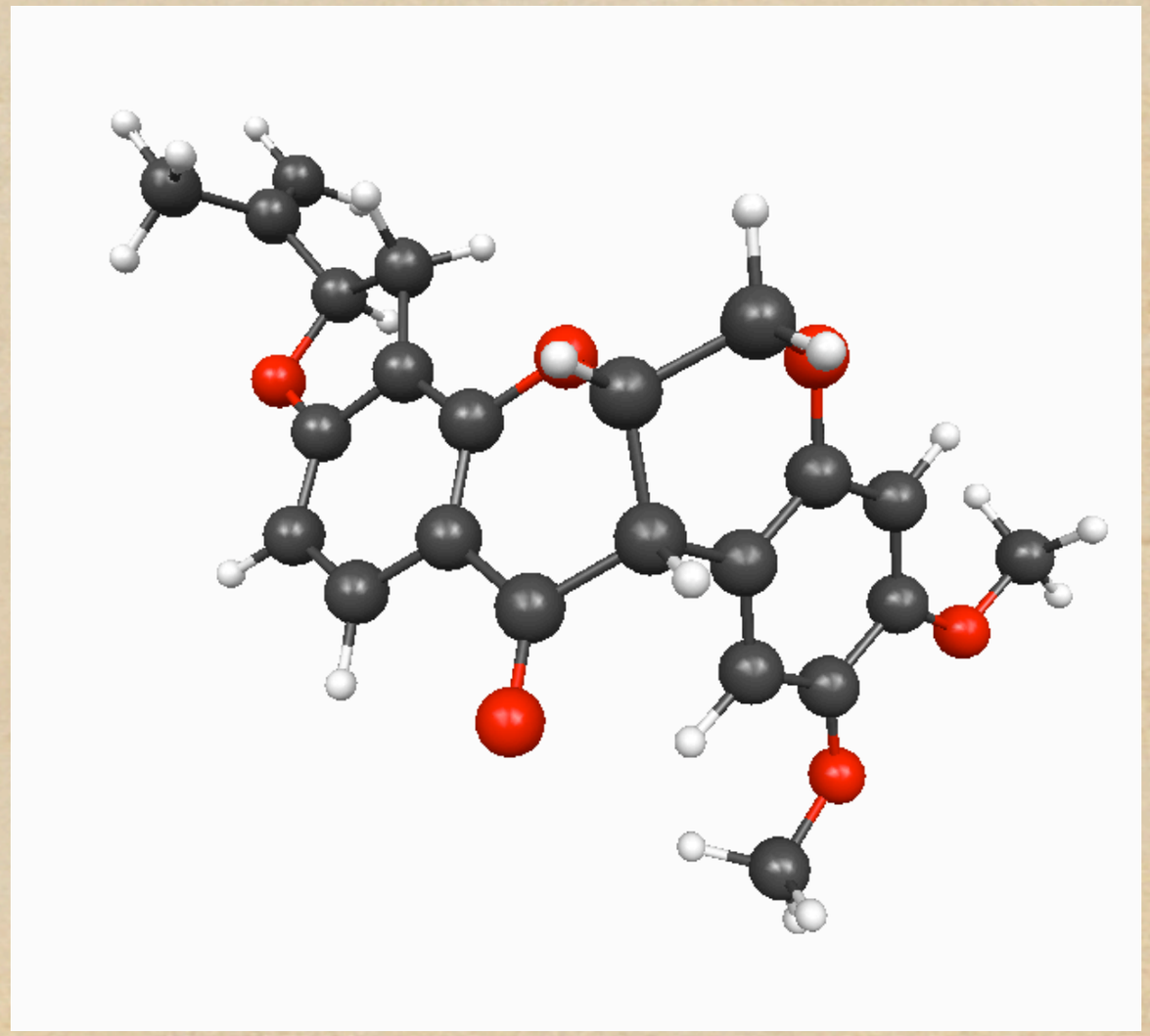
- Tool for measuring network bandwidth versus message size.
- New modules to test I/O bandwidth
  - Can be set to allow testing of file system cache (reread the same data over and over)
  - Can also stride through a file to obtain performance numbers all the way to disk.

GAMESS

- Our motivating application
- Large (750k lines) FORTRAN application
- Has many different algorithms including both direct (~diskless) and conventional (potentially very large temporary files).
- MPI version, but not normally used.
- Used the common Hartree-Fock energy calculation for our tests.

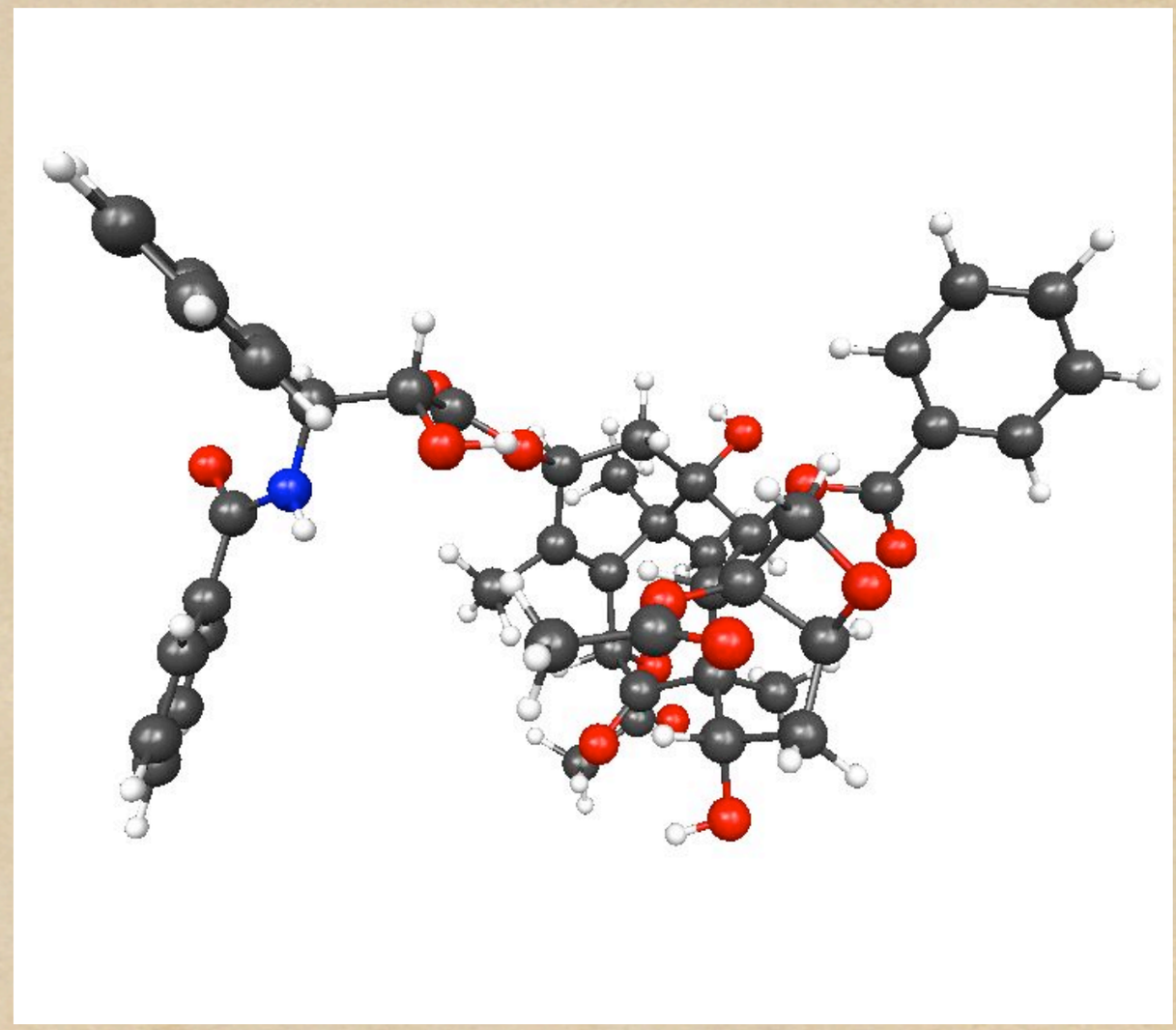
Small Test - Rotenone

- 479 AOs, 104 occupied MOs
- Produces 16.2 GB scratch file.



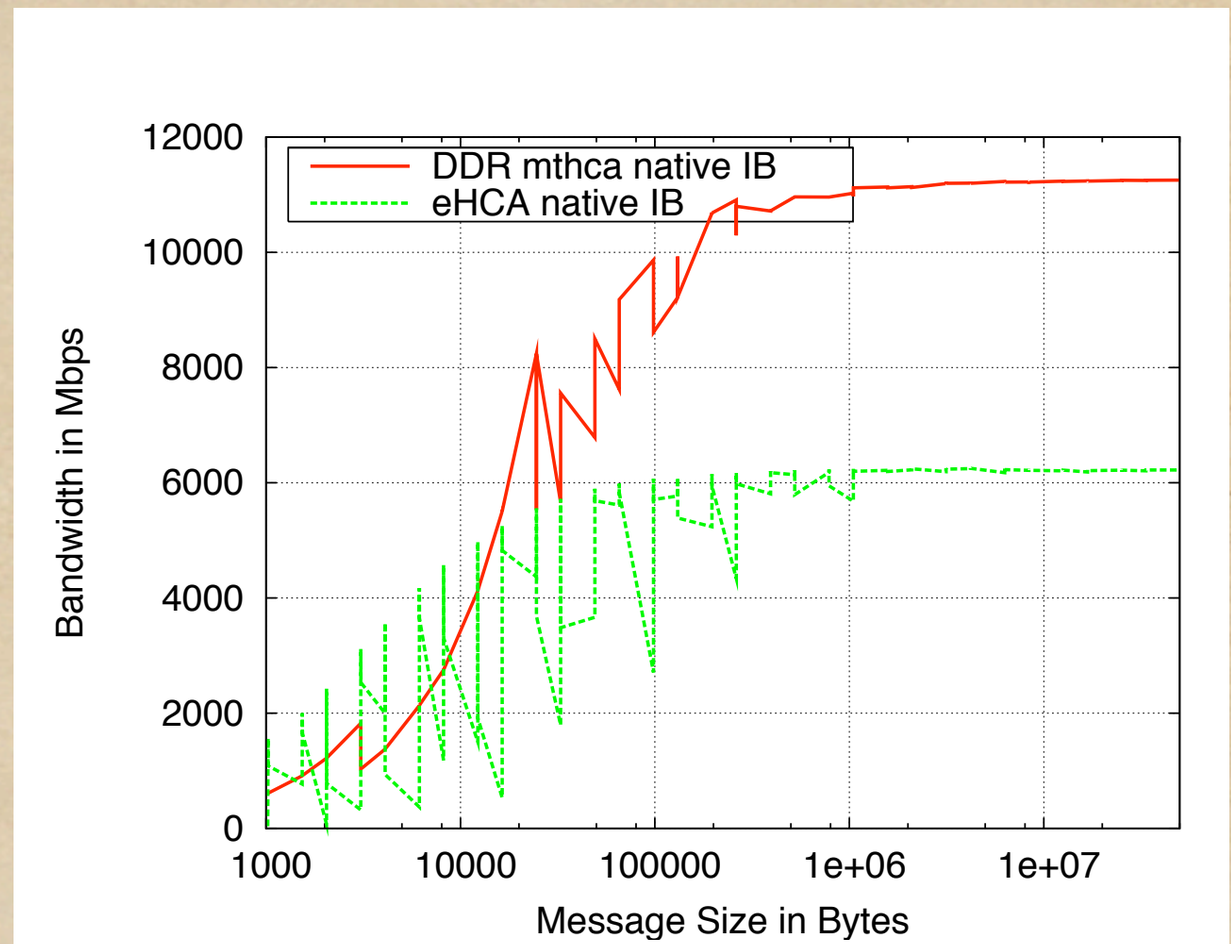
Large Test - Taxol

- 1032 AOs, 226 occupied MOs
- Produces 120 GB scratch file.



Base Network Performance

- Performance for the storage servers exceeds 11 Gbps
- IBM eHCA performance is a disappointing 6.2 Gbps.
  - eHCA has 6 DMA engines
  - eHCA can parallelize multiple streams with the multiple DMA engine

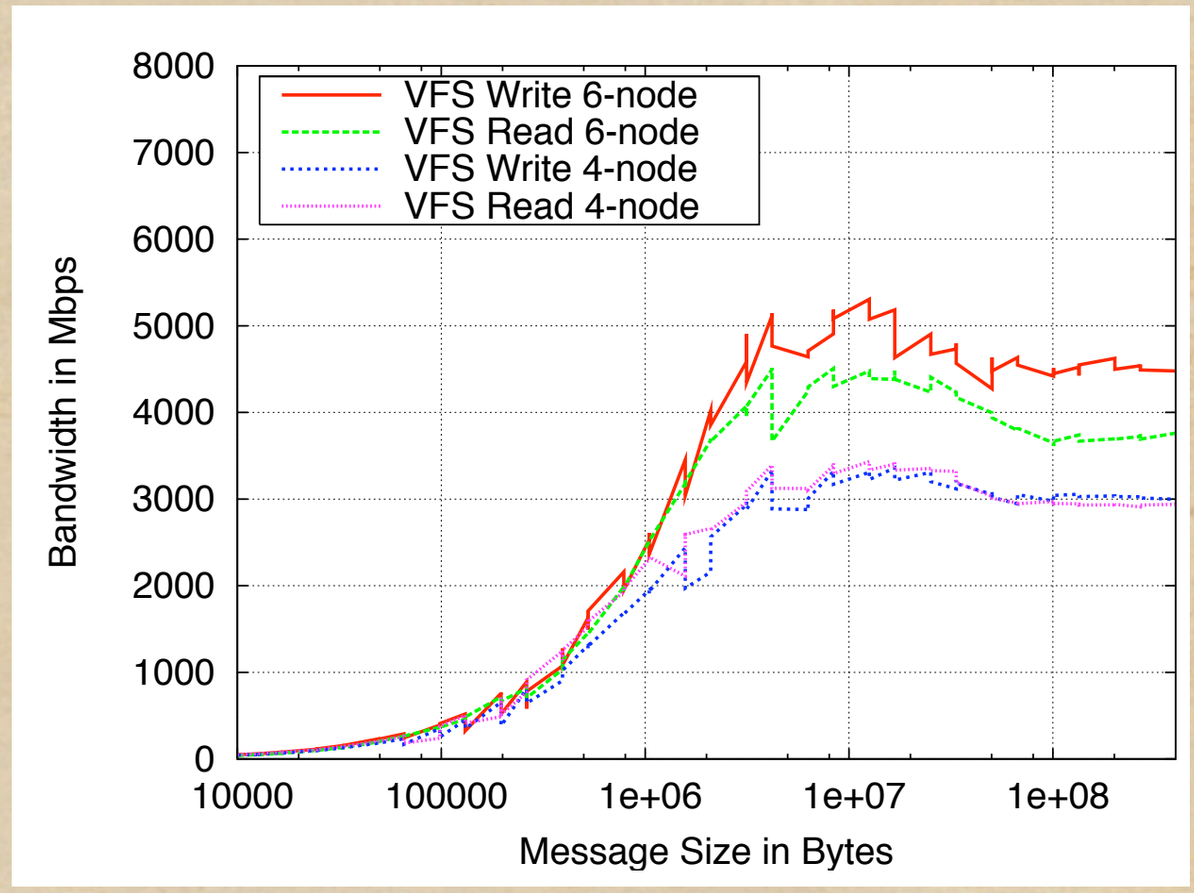


Base Disk Performance

- Directly testing of the I/O performance on the Opteron storage servers indicated a peak read performance of 435 MB/sec. measured using NetPIPE (a single stream). Much higher bandwidth can be obtained with Linux AIO approaching 600 MB/sec.

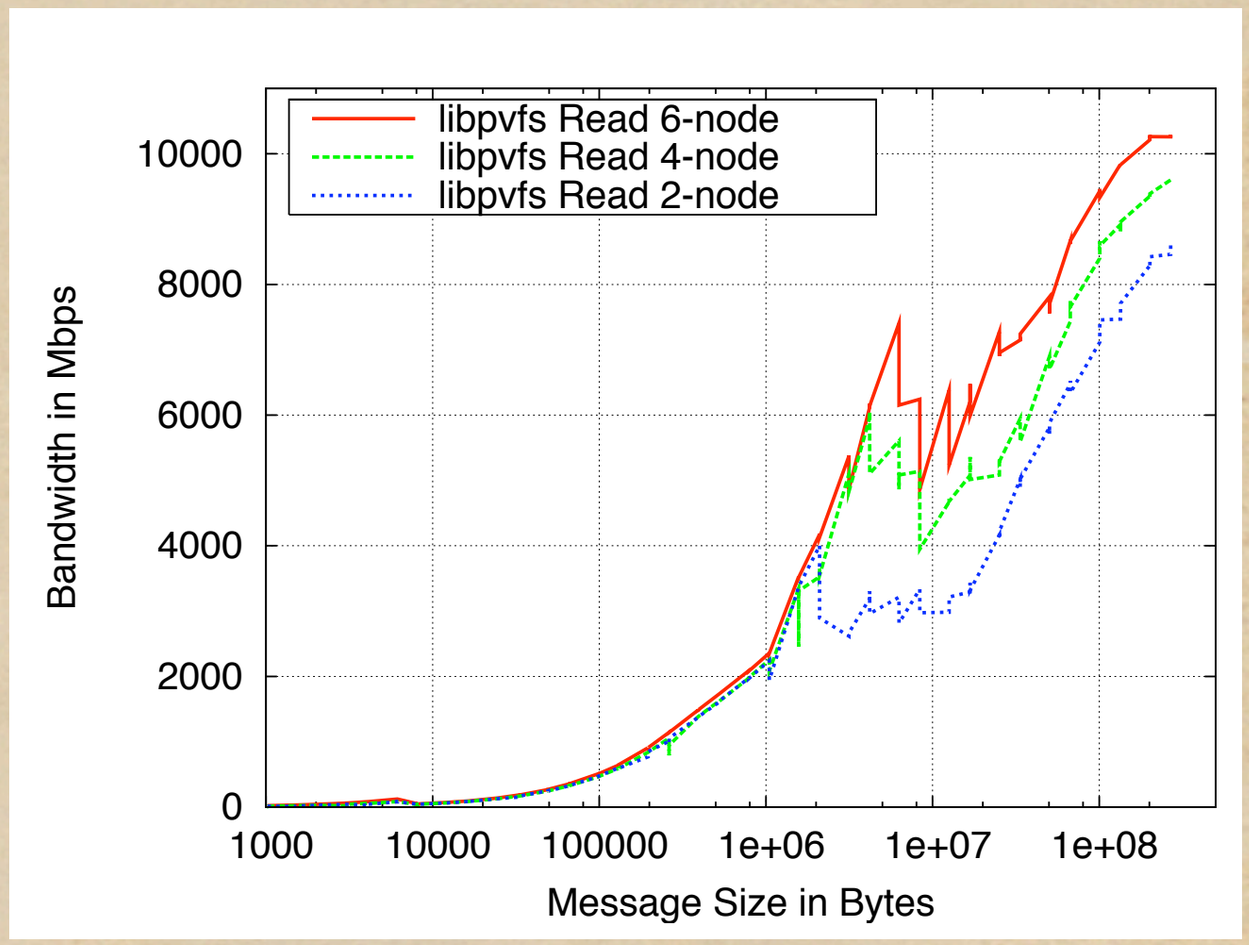
VFS Results from Cache

- Peak read performance of greater than 500 MB/sec
- GAMESS tests on small test case show similar peak numbers.



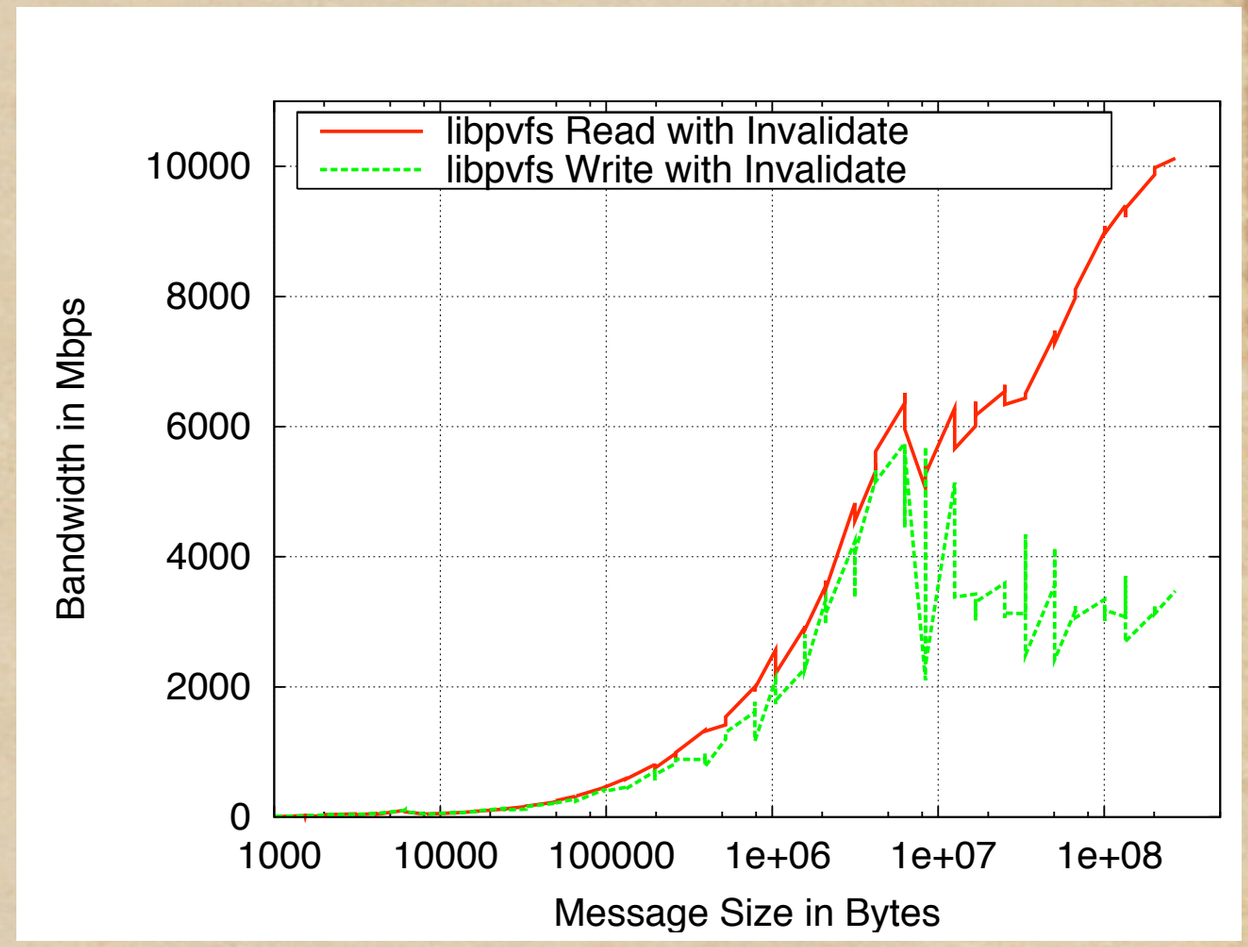
Native Results from Cache

- Peak read performance of greater than 1 GB/ sec
- GAMESS performance is also nearly double the VFS result.



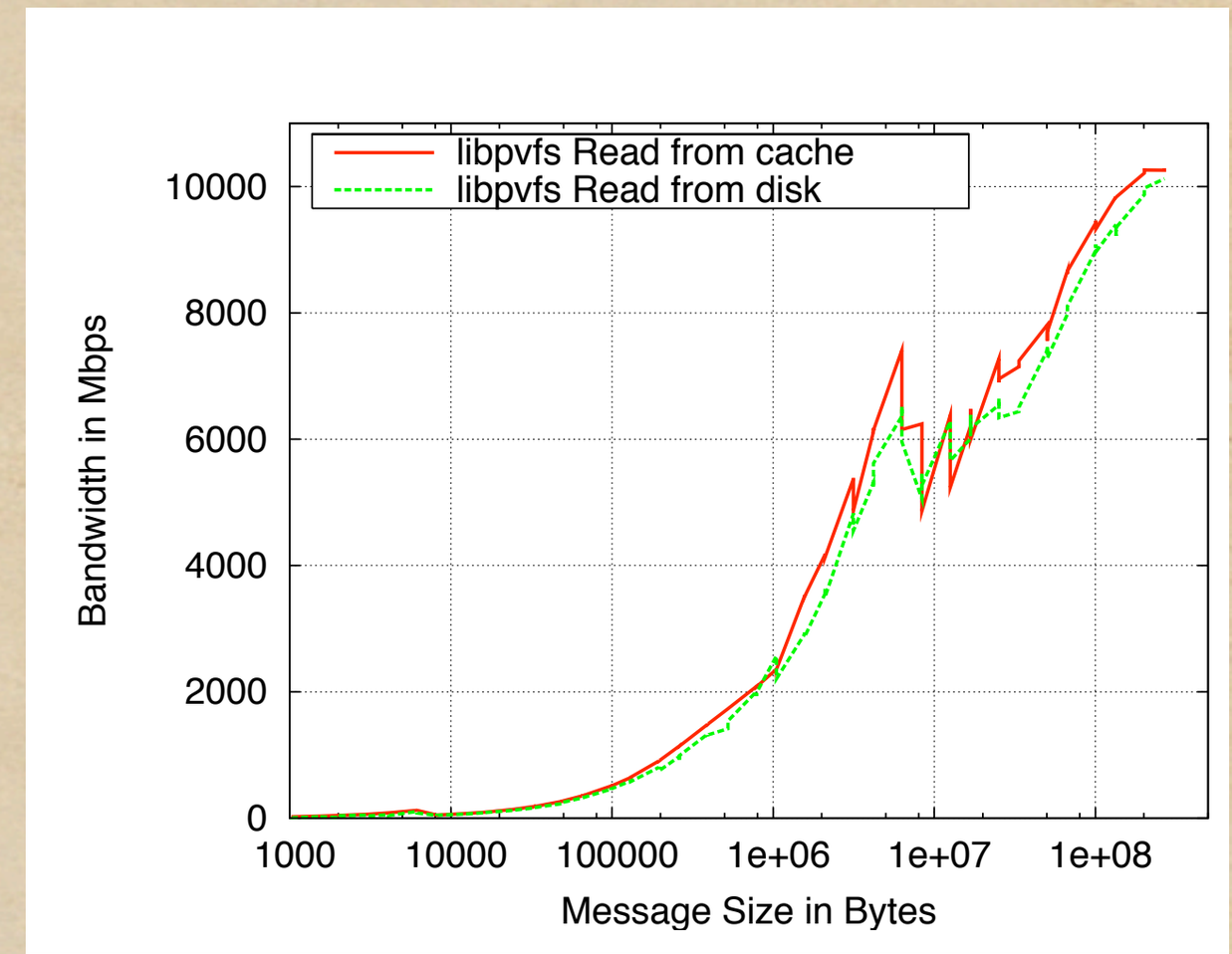
Results from Disk

- Write performance all the way to disk is significantly reduced.
- Read performance seems barely effected.



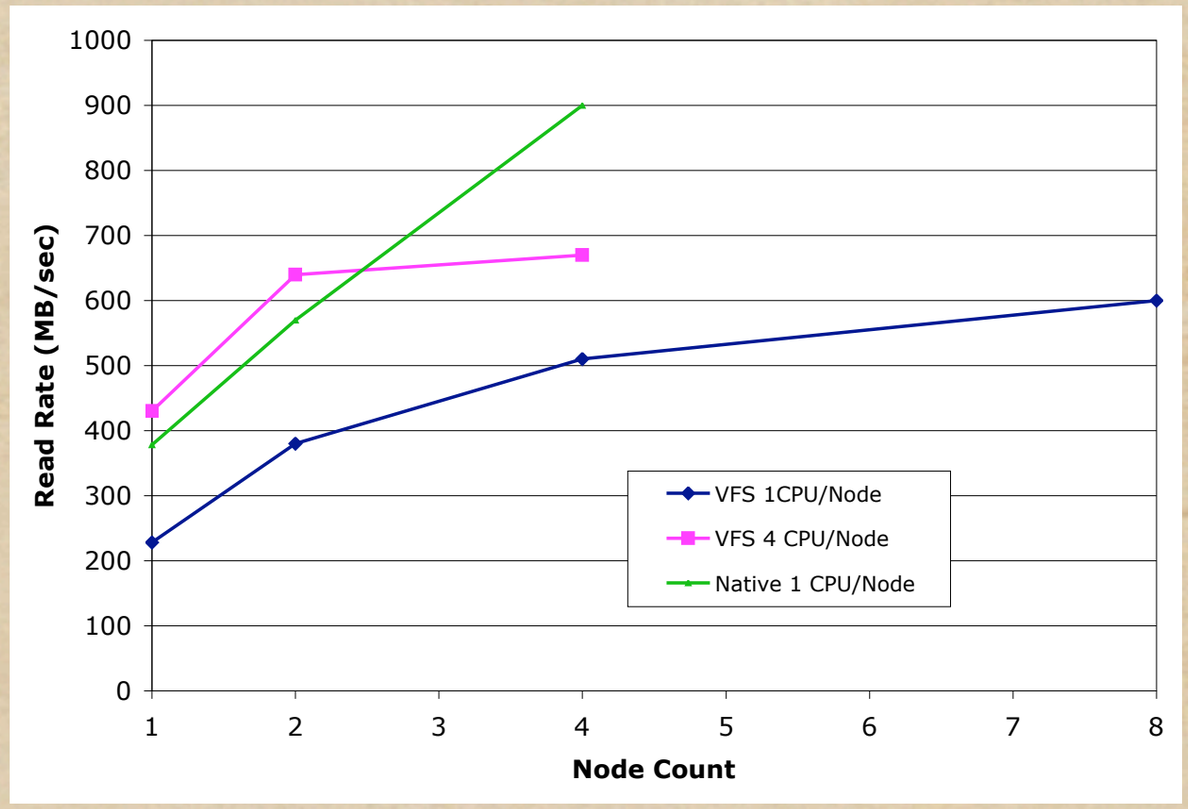
Results from Disk

- Indeed read performance is only slightly reduced with some additional latency.



GAMESS Large Run Performance

- Single process performance is 228 MB/sec over VFS, 375 MB/sec native.
- 4 processes on 4 nodes gives 900 MB/sec in native mode!



Conclusions

- PVFS2 over OpenIB can be used to deliver I/O to a single node and a single process at rates that significantly exceed the performance of locally attached disk subsystems typically used in clusters.
- This setup offers the possibility of using inexpensive storage servers to provide very fast I/O to high-end compute servers.

Acknowledgments

- Funding:
  - U.S. Department of Energy
  - IBM
- Brad Benton and Chet Mehta at IBM
- Pete Wyckoff at OSC
- Rob Latham, Sam Lang and Rob Ross on the PVFS2 development team.

Questions

?